PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : G07F 7/10 — A1

(11) International Publication Number: WO 00/68903

(43) International Publication Date: 16 November 2000 (16.11.00)

(21) International Application Number: PCT/US00/12933

(22) International Filing Date: 10 May 2000 (10.05.00)

(30) Priority Data:

60/133,447	11 May 1999 (11.05.99)	US
09/552,550	19 April 2000 (19.04.00)	US

(71) Applicant: MICROSOFT CORPORATION [–/US]; One Microsoft Way, Redmond, WA 98052–6399 (US).

(72) Inventors: DEO, Vinay; 15606 N.E. 40th Street #G225, Redmond, WA 98052 (US). MILSTEIN, David; 6610 159th Avenue N.E., Redmond, WA 98052 (US). PERLIN, Eric, C.; 845 Bellevue Place East, Apt. 302, Seattle, WA 98102 (US). ODINAK, Gilad; 12342 N.E. 26th Place, Bellevue, WA 98005 (US). GUTHERY, Scott, B.; 80 Manemet Road, Newton, MA 02459–1451 (US).

(74) Agents: BANOWSKY, James, R. et al.; 421 W. Riverside Avenue, Suite 500, Spokane, WA 99201 (US).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published

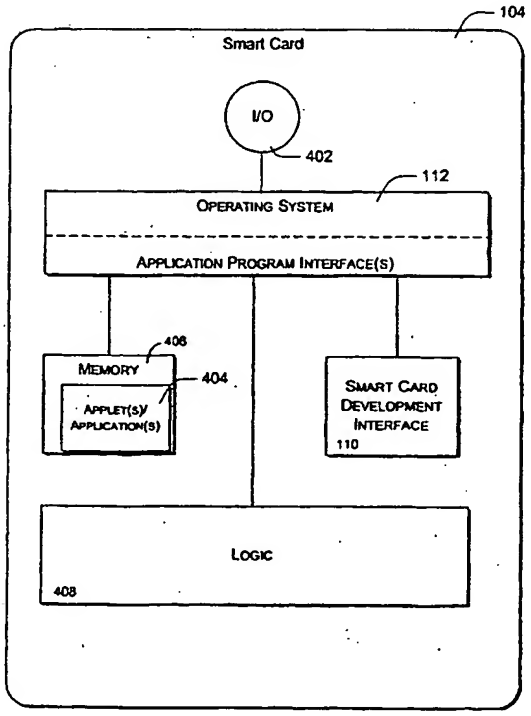
With international search report.

Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: SMART CARD APPLICATION DEVELOPMENT SYSTEM AND METHOD

(57) Abstract

A smart card includes a smart card development interface (SCDI) to receive remote procedure calls from a computer system to application program interface (API) features of the smart card operating system, to invoke the API called in the received RPC, and to respond to the computer system as appropriate.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

Smart Card Application Development System and Method

TECHNICAL FIELD

This invention relates to a class of devices commonly known as smart cards and, in particular, to an application development system and method for smart cards.

BACKGROUND OF THE INVENTION

Today there is increasing use of "smart cards" in place of, or in addition to, conventional magnetic stripe cards ("mag cards"). A "smart card" is a thin card embedded with a memory device (volatile and/or non-volatile) and associated programmable or non-programmable logic. Unlike the mag card that merely stores "static" information (e.g., a credit card account number), a smart card can add, delete and otherwise manipulate information stored on the card. Accordingly, smart cards are capable of storing and executing applications to carry out one or more functions within a smart card.

While the physical dimensions and processing features of the smart card give rise to potentially limitless applications, the reality is that smart card applications are only being developed for large scale markets, e.g., banking, security and transportation applications. One reason for this limited growth lies in the cost associated with smart card application development. There are several reasons why smart card application development is a costly undertaking, not the least of which is the "closed" nature of the smart card and the limited processing, memory and input/output resources of the smart card.

A smart card is often referred to as a "closed" system because, for security purposes, a smart card is purposefully designed to not expose its memory, intermediate system states or data and address bus information to external devices. To do so would render it susceptible to unauthorized access (hacking) and fraud. While its closed nature is useful for secure applications such as banking transactions, it makes it difficult to utilize prior art smart cards for development purposes. It is to be appreciated that application development often requires access to memory or bus values, or system state information during intermediate processing steps, access that has been specifically designed out of the smart card.

Another encumbrance to the smart card application designer is the limited resources of the smart card. That is, due to the physical and processing constraints placed on the smart card, prior art smart cards do not enjoy any dedicated debug facilities. Aside from the limited processing and memory attributes of a smart card, a smart card typically has but a single, bi-directional input/output (I/O) port. The communication bandwidth of this single I/O port is typically consumed to support execution of the smart card application itself, leaving little to no communication bandwidth to support debug features. Thus, application development using a smart card itself is virtually impossible. Consequently the development of applications for a smart card currently requires the use of an in-circuit emulator (ICE) and an associated, often proprietary software development application.

An ICE system is typically comprised of a printed circuit card coupled to a computer system executing a proprietary software development application associated with the printed circuit card (emulator). The printed circuit card is designed to emulate the functionality of the smart card, while providing additional debug facilities (e.g., I/O ports, memory buffers, address and data lines and the like), thereby providing the developer with the necessary access to adequately

debug their applications in development. One limitation of such smart card development systems is that the ICE and proprietary development application are chip-specific. Thus, an emulator for smart card employing a Siemens processor will not work with an emulator employing a Philips or Motorola processor without significant hardware modification. Moreover, the software development application executing on the computer system is also chip-specific, with an associated chip-specific compiler, linker and debugger, and often require that a developer learn the "programming language" of the development tool. Consequently, an application developed on one ICE system cannot be utilized (or directly ported to) a smart card employing a different processor without costly modification.

These emulators work with special versions of the smart card processors called 'bondouts' – these are the same processors as in the smart card but they expose their memory, data bus, etc to facilitate debugging. Since these special processors behave exactly like the processor in the smart card, many smart card processor manufacturers consider them to be a high risk items that may allow hackers to learn the smart card processor's deficiencies. To prevent this they screen recipients and make them sign restrictive agreements before these bondouts can be supplied. This model of working is totally unsuitable for the large scale acceptance and use by existing PC developer community.

As a result of each of the foregoing limitations, smart card application development is a costly undertaking, typically performed by the large corporations that stand to profit from the sale of millions of smart cards. History has shown that in order for a new technology to blossom, "grass roots" application development is required. That is, a technology will not truly become a pervasive technology unless

and until it is infused with the vitality and creativity of individual programmers and small development companies.

Thus, an improved application development environment is required for smart card applications that is unencumbered by the limitations commonly associated with the prior art. One such solution is presented below.

SUMMARY OF THE INVENTION

This invention concerns an integrated circuit (IC) card, such as a smart card and, more particularly, an improved application development system for smart card applications. In accordance with a first aspect of the invention, a smart card includes a smart card development interface (SCDI) to receive remote procedure calls from a computer system to application program interface (API) features of the smart card operating system, to invoke the API called in the received RPC, and to respond to the computer system as appropriate.

According to another aspect of the invention, a computer system includes an innovative client development interface (CDI), to marshal and issue a remote procedure call to an associated API of the operating system of the smart card, and to receive and present a response from the smart card for presentation to a calling application, as appropriate.

It is to be appreciated that combination of the foregoing aspects of the present invention gives rise to a smart card application development system comprising a computer system endowed with the client development interface (CDI), and a smart card incorporating the smart card development interface, wherein the computer system is executing a common development application that issues commands to smart card API's, whereupon the CDI intercepts the remote procedure calls, marshals the parameters associated with the command and issues

the command to the smart card. The development interface of the smart card receives and un-marshals the command and invokes an API called by the RPC, marshaling and issuing a response to the computer system as appropriate. A development system incorporating these innovative aspects facilitates smart card development without the need for an ICE system, typical of the prior art. Thus, the present invention represents a new paradigm in smart card application development, enabling entry into the market of individual programmers and small development companies.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a block diagram of an example smart card application development system including a computer system and a smart card;

Fig. 2 is a block diagram of an example computer system including an innovative client development interface, suitable for use in the smart card application development system of Fig. 1;

Fig. 3 is a block diagram of an example client development interface suitable for use within the computer system of Fig. 2;

Fig. 4 is a block diagram of an example smart card including a smart card development interface, suitable for use in the smart card application development system of Fig. 1;

Fig. 5 is a block diagram of an example smart card development interface, suitable for use within the smart card of Fig. 4;

Fig. 6 is a graphical illustration of a data structure containing an example proxy library suitable for use in the computer system of Fig. 3;

Fig. 7 is a graphical representation of an example marshaling buffer suitable for use with the present invention;

Fig. 8 is a graphical representation of an example unmarshaling buffer suitable for use with the present invention; and

Fig. 9 is a flow chart of an example method for smart card application development utilizing the innovative smart card application development system depicted in Fig. 1.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Example Development System

Fig. 1 illustrates a smart card application development system 100 comprising a computer system 102 employing a generic software development application coupled to a smart card 104. The exemplary development system 100 of Fig. 1 depicts computer system 102 coupled to smart card 104 via a card reader 106 and a communication medium 108. The communication medium 108 is intended to represent any of a number of typical communication links including, but not limited to, a proprietary data bus, an industry standard data bus, a local area network (LAN), a wide area network (WAN), or a global area network (e.g., the Internet). In this regard, the innovative smart card application development system 100 facilitates remote development of smart card applications using an actual smart card 104 coupled to a remote computer system 102 via a communications network 108 and a card reader 106.

According to one aspect of the present invention, smart card 104 comprises a smart card development interface 110 coupled to an operating system (OS) 112 with a set of innovative smart card application program interfaces (APIs). It is to be appreciated that the APIs provide a wide range of functional capability within the smart card, albeit limited to the application area suitable for smart cards . As will be developed in greater detail below, the smart card development interface (SCDI)

110 receives remote procedure calls (RPC) from computer system 102 and selectively invokes APIs associated with operating system 112 enabling smart card 104 to carry out specific functional tasks. In this way, the functionality of innovative smart card 104 is controlled in a step-wise fashion that is conducive to application development. It is to be appreciated that, but for the smart card development interface (SCDI) 110, smart card 104 and its operating system with associated API's 112 is intended to represent any of a broad range of integrated circuit cards and their operating systems commonly known in the art. That is, any smart card endowed with the smart card development interface 110 is suitable for use within the smart card application development system 100 of Fig. 1.

It is noted that, in addition to the illustrated smart cards, the IC card might be embodied in other forms, such as an electronic wallet, a personal digital assistant, a smart diskette (i.e., an IC-based device having a form factor and memory drive interface to enable insertion into a floppy disk drive), a PC card (formerly PCMCIA card), and the like. Generally, the integrated circuit card 104 is characterized as an electronic device with limited processing capabilities and memory wherein large size number crunching is either absent or is provided for specific security purposes (e.g., to execute cryptographic algorithms). For purposes of this discussion and within the context of the illustrated implementation, the terms "IC device", "IC card", and "smart card" will be used interchangeably to reference the smart card 104.

Card reader 106 provides a necessary interface between smart card reader 104 and a computing system such as, e.g., computer system 102. Card readers are typically designed to support any of a number of standardized communication protocols supported within the smart card community and, in this way, can typically accommodate smart cards adhering to any of the recognized communication

standard from any smart card manufacturer. In this regard, card reader 106 is typically not chip- or card-specific although in some cases it may be. For purposes of this discussion, card reader 106 includes the necessary hardware and software resources required to support standardized communication between computer 102 and smart card 104. Consequently, card reader 106 is merely intended to be illustrative of card readers typically known within the art.

Computer system 102 is depicted within Fig,. 1 as comprising an innovative client development interface (CDI) 114, a plurality of executable applications 116 including a development application 118, each of which is supported by the resources of a typical operating system 120, as shown. As will be developed in more detail below, the client development interface (CDI) 114 identifies remote procedure calls (RPCs) to smart card 104 resources (e.g., API's), marshals the parameters required by the smart card 104 and issues the RPC to the smart card development interface 110 via communication channel 108 and card reader 106. The smart card 104 receives and executes the called API and returns the result to the calling application via the communication channel 108 and CDI 114, as appropriate. But for the client development interface 114, computer system 102, applications 116 and operating system 120 are each intended to represent any of a number of commonly known computer systems, applications and operating systems, respectively, known in the art.

According to one aspect of the present invention, any of a number of prior art development applications, or tools, may well be used in accordance with the innovative SCDI 110 and CDI 114 of development system 100 to develop smart card applications. Examples of such software development tools include Visual Basic or Visual C/C++ from Microsoft Corporation of Redmond, WA. Thus, a smart card application is developed using computer 102 and a typical software

development tool 118, utilizing smart card API calls within the developed code. The smart card API's are recognized by client development interface 114, described above, which selectively invokes the called resources of the smart card 102.

In this regard, development system 100 comprising an innovative SCDI 110 and a CDI 114 effectively liberates a developer from the costly and cumbersome proprietary development tools commonly associated with the prior art.. Rather the present invention facilitates application development using a standard personal computer 102 endowed with typical software development tools and an innovative client development interface 114 can develop smart card applications using smart card 104 endowed with a smart card development interface 110.

Example Computer System

In the discussion herein, the invention is described in the general context of computer-executable instructions, such as program modules, being executed by one or more conventional computers. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Moreover, those skilled in the art will appreciate that the invention may be practiced with other computer system configurations, including hand-held devices, personal digital assistants, multiprocessor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, and the like. In a distributed computer environment, program modules may be located in both local and remote memory storage devices.

Fig. 2 shows a general example of a computer system 102 incorporating the teachings of one aspect of the present invention, and suitable for use within the smart card application development system 100. It will be evident, from the

discussion to follow, that computer 102 is intended to represent any of a class of general or special purpose computing platforms which, when endowed with the innovative client development interface, is suitable for use in smart card application development system 100. In this regard, the following description of computer system 102 is intended to be merely illustrative, as computer systems of greater or lesser capability may well be substituted without deviating from the spirit and scope of the present invention.

As shown, computer 102 includes one or more processors or processing units 132, a system memory 134, and a bus 136 that couples various system components including the system memory 134 to processors 132.

The bus 136 represents one or more of any of several types of bus structures, including a memory bus or memory controller, a peripheral bus, an accelerated graphics port, and a processor or local bus using any of a variety of bus architectures. The system memory includes read only memory (ROM) 138 and random access memory (RAM) 140. A basic input/output system (BIOS) 142, containing the basic routines that help to transfer information between elements within computer 102, such as during start-up, is stored in ROM 138. Computer 102 further includes a hard disk drive 144 for reading from and writing to a hard disk, not shown, a magnetic disk drive 146 for reading from and writing to a removable magnetic disk 148, and an optical disk drive 150 for reading from or writing to a removable optical disk 152 such as a CD ROM, DVD ROM or other such optical media. The hard disk drive 144, magnetic disk drive 146, and optical disk drive 150 are connected to the bus 136 by a SCSI interface 154 or some other suitable bus interface. The drives and their associated computer-readable media provide nonvolatile storage of computer readable instructions, data structures, program modules and other data for computer 102. Although the exemplary environment

described herein employs a hard disk 144, a removable magnetic disk 148 and a removable optical disk 152, it should be appreciated by those skilled in the art that other types of computer readable media which can store data that is accessible by a computer, such as magnetic cassettes, flash memory cards, digital video disks, random access memories (RAMs) read only memories (ROM), and the like, may also be used in the exemplary operating environment.

A number of program modules may be stored on the hard disk 144, magnetic disk 148, optical disk 152, ROM 138, or RAM 140, including an operating system 158, one or more application programs 160 including, for example, the innovative client development interface 114, other program modules 162, and program data 164. A user may enter commands and information into computer 102 through input devices such as keyboard 166 and pointing device 168. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are connected to the processing unit 132 through an interface 170 that is coupled to bus 136. A monitor 172 or other type of display device is also connected to the bus 136 via an interface, such as a video adapter 174. In addition to the monitor 172, personal computers often include other peripheral output devices (not shown) such as speakers and printers.

As shown, computer 102 operates in a networked environment using logical connections to one or more remote computers, such as a remote computer 176. The remote computer 176 may be another personal computer, a personal digital assistant, a server, a router or other network device, a network "thin-client" PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to computer 102, although only a memory storage device 178 has been illustrated in Fig. 2.

As shown, the logical connections depicted in Fig. 2 include a local area network (LAN) 180 and a wide area network (WAN) 182. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets, and the Internet. In one embodiment, remote computer 176 executes an Internet Web browser program such as the "Internet Explorer" Web browser manufactured and distributed by Microsoft Corporation of Redmond, Washington to access and utilize online services.

When used in a LAN networking environment, computer 102 is connected to the local network 180 through a network interface or adapter 184. When used in a WAN networking environment, computer 102 typically includes a modem 186 or other means for establishing communications over the wide area network 182, such as the Internet. The modem 186, which may be internal or external, is connected to the bus 136 via a serial port interface 156. In a networked environment, program modules depicted relative to the personal computer 102, or portions thereof, may be stored in the remote memory storage device. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

Generally, the data processors of computer 102 are programmed by means of instructions stored at different times in the various computer-readable storage media of the computer. Programs and operating systems are typically distributed, for example, on floppy disks or CD-ROMs. From there, they are installed or loaded into the secondary memory of a computer. At execution, they are loaded at least partially into the computer's primary electronic memory. The invention described herein includes these and other various types of computer-readable storage media when such media contain instructions or programs for implementing the innovative steps described below in conjunction with a microprocessor or other data processor.

The invention also includes the computer itself when programmed according to the methods and techniques described below. Furthermore, certain sub-components of the computer may be programmed to perform the functions and steps described below. The invention includes such sub-components when they are programmed as described. In addition, the invention described herein includes data structures, described below, as embodied on various types of memory media.

For purposes of illustration, programs and other executable program components such as the operating system are illustrated herein as discrete blocks, although it is recognized that such programs and components reside at various times in different storage components of the computer, and are executed by the data processor(s) of the computer.

Example Client Development Interface

Fig. 3 illustrates a block diagram of an example client development interface 114, suitable for use in computer system 102. As shown, client development interface 114 comprises control logic 302, an application program interface proxy library 304 and one or more marshaling/un-marshaling buffers 306, each coupled as depicted. According to this implementation, client development interface 114 is invoked by a calling application. Thus, in order to utilize the resources of CDI 114 and, more specifically, the proxy library 304, the CDI 114 must be identified within the executing application. In one embodiment, this is done by identifying the resource in the code of the application.

Control logic 302 is intended to represent any of a broad range of logic known in the art. In one implementation, control logic 302 is a processor, while in alternate embodiments control logic is a microcontroller, a programmable logic array, or a series of executable instructions which perform logic functions. Control

logic 302 communicates with smart card 104 in any of a plurality of standard smart card communication protocols. In alternate embodiments, non-standard protocols may well be used to communicate between controller 302 and smart card 104 such as, for example, a unique development communication protocol. Although not specifically denoted, it is to be appreciated that controller 302 communicates with smart card 104, and any other peripheral for that matter, via the communication resources of operating system 120. Insofar as such resources are well known in the art, they need not be described further here.

As used herein, "marshaling" means converting a function argument from its binary representation (i.e., compiled application format) into some language-independent format, and then converting this generic representation into a binary format appropriate to the called function, (i.e., that of the smart card). Accordingly, the marshaling/unmarshaling buffer(s) 306 are one or more buffers wherein the parameters necessary to issue a complete API call, as defined by the proxy library, are compiled before issuance to the SCDI 110 of the smart card 104. Accordingly, marshaling/un-marshaling buffer(s) 306 are intended to represent any of a number of alternate memory devices commonly known to those in the art.

The application program interface proxy library 304 provides a functional library of smart card application program interfaces available to a calling application. According to one implementation, proxy library 304 contains a detailed listing of all smart card API's, associated function calls and function parameters necessary to invoke the API on smart card 104. As will be developed in more detail below, upon recognizing a call to a smart card API, control logic 302 of CDI 114 accesses proxy library 304 to identify the associated function call and required parameters necessary to properly invoke the function on the smart card 104. Control logic 302 establishes a buffer entry within marshaling/un-marshaling

buffer(s) 306 in which to marshal (or compile) the necessary parameters. Once the necessary parameters have been collected (marshaled), control logic 302 issues a command to the smart card 104 via network 108 and card reader 106 using a smart card standard communication protocol such as, for example, International Standards Organisation (ISO) 7816 T=0, or T=1.

Although depicted as a separate functional element, those skilled in the art will appreciate that client development interface 114 may well be integrated within and utilize the control features associated with the software development application 118. In one implementation, for example, control logic 302 and the marshaling/un-marshaling buffer(s) 306 may well be supplied by a debug environment within the application development tool 118, wherein client development interface is comprised solely of proxy library 304. Accordingly, the teachings of the present invention may well be practiced with variation from the exemplary embodiment without deviating from the spirit and scope of the present invention.

Application Program Interface

Those skilled in the art will appreciate that an application program interface, or API, interfaces a (usually) higher-level program such as an application to lower-level services and functions of another application or resource. In this regard, the API provides an "abstraction layer" to the functions and services of another resource. According to one aspect of the present invention, proxy library 304 provides a number of innovative IC card API's that enable a developer to write and test smart card applications on a host computer system (102) using the functional resources of a communicatively coupled IC card (104).

According to this aspect, the application running on the host can use the remote procedure call (RPC) method, introduced above, to access the APIs on the card. The API calls in such programs are identified by control logic 302 of CDI 114 and *proxied* to SCDI 110 of the IC card 104 via marshaling buffer 306. The API itself is thus executed on the card 104 and the results are sent back to the application executing on computer 102 via SCDI 110, communication channel 108 and CDI 114, respectively. As alluded to above, to utilize CDI 114 and proxy library 304, applications need to include a header file and link to the proxy library. In one implementation, the header file is the "wincard-proxy.h" header file and the scwapi.dll API library. After referencing such files in the header, a call to a smart card resource will be in the form:

SCODE WINAPI API Call (API arguments) (1)

For purposes of illustration, and not limitation, some example API's are presented in Table I, below, that enable an application executing on a computer system (102) to utilize the functional resources of a communicatively coupled IC card (104), according to the teachings of the present invention. A more complete description of these and additional innovative APIs are provided in Appendix A.

Function Call	Description
ScwAttachToCard	A prerequisite for all other RPC API calls; used to connect with the card and initialize internal data structures that are needed for other API calls.
ScwDetachFromCard	Frees the resources of an identified IC card; this API should always be called at the end of an RPC

	session.
ScwCreateFile	Enables an application executing on a computer to open an existing file residing on the IC card, or to create one if it does not already exist.
ScwDeleteFile	Enables an application executing on a computer to delete the indicated file, which could be an access-control list (ACL) or a directory.
ScwWriteFileByName	Enables an application to write to an IC card file at an indicated offset without having to open the file.
ScwReadFileByName	Enables an application to read from an IC card file at an indicated offset without having to open the file.
ScwIsAuthenticated	Enables an application to invoke an authentication function on the IC card.

Table I: Example IC Card APIs

It will be appreciated that the RPC development environment facilitated by the innovative APIs, SCDI 110 and CDI 114 execute with all of the speed and resources of the host computer, while eliminating the need to "emulate" the smart card resources – an actual smart card is utilized instead.

Example Smart Card

Fig. 4 illustrates a block diagram of an example smart card 104 suitable for use within smart card application development system 100 of Fig. 1. In addition to the innovative smart card development interface (SCDI) 110 and an operating system with associated APIs 112, smart card 104 is shown comprising an

input/output interface 402, a plurality of applications 404 including smart card development interface 110, memory 406 and control logic 408. As discussed above, except for the inclusion of innovative smart card development interface (SCDI) 110, to be described more fully below, smart card 104 is intended to represent any of a broad category of integrated circuit (IC) cards commonly known in the art. Thus, but for SCDI 110, each of I/O 402, applications 404, memory 406 and control logic 408 are likewise commonly known within the art and, consequently, will not be further described here.

Fig. 5 illustrates a block diagram of an example smart card development interface (SCDI) 110, suitable for use within smart card 104 of application development system 100 in Fig. 1. In accordance with the example implementation of Fig. 5, SCDI 110 is shown comprising control logic 502 and marshaling/un-marshaling buffer(s) 504, coupled as depicted. According to one implementation, SCDI 110 is invoked by CDI 114 upon the initial remote procedure call to smart card 104.

According to this exemplary implementation, control logic 502 may be any of a plurality of logic and logic functions commonly known in the art such as, for example, a processor, a controller, or a plurality of executable instructions which implement such functionality. Control logic 502 receives a function call from CDI 114 via communication channel 108, card reader 106 and smart card I/O 402 according to any of a plurality of communication protocols, described above. In response, control logic 502 invokes the called API, utilizing the parameters received in the marshaled function call. Smart card 104 executes the called API. If a response is called for by the executing API, control logic 502 establishes a buffer entry within marshaling/un-marshaling buffer(s) 504 and, once marshaled, issues the response to the CDI 114 via smart card I/O 402, card reader 106 and

communication channel 108. Accordingly, control logic 502 receives and interprets the remote procedure calls from the CDI 114 to selectively invoke called API's of the smart card OS 112, and to return response information to CDI 114 as appropriate.

As described above, SCDI 110 is responsive to the application development interface of an appropriately endowed computer system, e.g., computer 102, to convert the otherwise ordinary smart card 104 into an essential development tool. No longer does a developer have to rely on emulation or simulation of smart card resources, but rather, the SCDI 110 opens the otherwise closed resources of the smart card 104 to the developer.

Example Data Structures

Fig. 6 is a graphical illustration of a data structure containing an example proxy library 600, suitable for use by computer system 102 of development system 100 of Fig. 1. As shown, proxy library 600 contains a list of remote procedure calls 602 corresponding to smart card application program interface (API) function codes 604. In addition, proxy library 600 includes fields for the size 606 of the function call, the number of parameters 608, the parameter type 610 and whether invocation of the function call elicits a response 612.

Fig. 7 is a graphical representation of an example send buffer suitable for use in association with the present invention. As shown, send (or marshalling) buffer 306 provides enough fields to accommodate elemental aspects of a smart card communication protocol. In accordance with the illustrated example embodiment of Fig. 7, marshaling buffer 306 is established by control logic 302 with fields 702 in accordance with the International Standards Organisation standard 7816 (ISO 7816) communication protocol command structure. As

described above, the marshaling buffer 306 is populated with information retrieved from proxy library 600.

Fig. 8 is a graphical illustration of an example response buffer suitable for use in association with the present invention. As shown, response buffer 504 provides enough fields to accommodate the elemental aspects of a smart card communication protocol. In accordance with the illustrated example embodiment of Fig. 8, response buffer 504 is established by control logic 502 to include a data_field, and two status byte fields (SW1, SW2), cumulatively referenced as 802. It is to be appreciated that although proxy library 600 is depicted as a two-dimensional data structure, this is for ease of explanation only. Data structures of greater or lesser complexity are anticipated within the scope of the present invention.

Example Operation

Fig. 9 is a flow chart of an example method for smart card application development utilizing the innovative smart card application development system depicted in Fig. 1. For ease of explanation, and not limitation, the method of Fig.9 will be developed with continued reference to Fig.'s 1-8.

With reference to Fig. 9, the method begins with step 902 wherein a developer writes a smart card application in any of a number of alternative languages and/or common application development tools, specifying a smart card proxy library and embedding remote procedure calls to smart card API's within the code of the application. As described above, the software development application 118 is immaterial, so long as the smart card proxy library is specified within application, CDI 110 will be invoked when the host computer 102 executing the

application processes an RPC to a smart card API. In step 904, the application is invoked.

In step 906, the application executes until a remote procedure call (RPC) to an IC card API is encountered. In step 908, upon detecting a smart card RPC, control logic 302 of CDI 114 accesses proxy library 304 to identify the functions and parameters associated with the RPC and establishes a buffer entry within marshaling buffer 306 to compile the required parameters. The control logic 302 prepares the buffer entry (also referred to as the send buffer) in marshaling buffer 306. To further explain this point, consider the example API call and the example proxy library of Fig. 6.

As described above, the information contained in the proxy library 600 enables CDI 106 to create a marshaling (or, send) buffer (306) to generate the API call to the smart card. In accordance with the illustrated example of Fig. 6, the ScwIsAuthenticated remote procedure call 614 is depicted, with associated information in corresponding fields 604-612. Accordingly, upon receiving the ScwIsAuthenticated RPC 614, CDI 106 creates a marshaling buffer 306 according to the information retrieved from the proxy library 600 to marshal the information required for transmission to the smart card. An example of the send buffer for the ScwIsAuthenticated is illustrated with reference to Fig. 7. It is to be appreciated that different RPC's of greater or lesser complexity may have associated send-buffers with a corresponding greater or lesser complexity. Appendix A includes a listing of innovative remote procedure calls and their associated send-buffer's, according to one aspect of the present invention.

In step 910, the completed send-buffer is sent to smart card 104 via communication channel 108 and card reader 106, and CDI 114 awaits a response from the smart card, as appropriate.

In step 912, control logic 502 of smart card development interface 110 receives the send buffer via I/O port 402 and invokes a corresponding application program interface (API), establishing a response buffer in buffer(s) 504, as appropriate. In step 914, upon completion of the API command at smart card 104, control logic 502 of smart card development interface 110 marshals response parameters in response buffer 504, according to definition of the called API, and sends the response buffer to the CDI 114 via I/O port 402, card reader 106 and communication channel 108. In accordance with the illustrated example embodiment, an example response buffer 504 is graphically illustrated with reference to Fig. 8. As above, the response buffer of Fig. 8 is for illustrative purposes, as response buffers of greater or lesser complexity corresponding with API's of greater or lesser complexity are anticipated within the scope of the present invention.

In step 916, upon receiving a response buffer from SCDI 110, control logic 302 of CDI 114 promotes the response to the calling application 118.

It is to be appreciated that the innovative smart card development interface 110 transforms the otherwise closed system of smart card 104 into an application development tool. Moreover, the client development interface 114 transforms a common application development tool such as Microsoft's Visual BASIC, or Visual C/C++ into a smart card application development tool. Accordingly, the combination of the smart card development interface 110 and the client development interface 114 enable a developer to enter the smart card development market with minimal cost, thereby facilitating the development of applications for limited-sized markets and promoting the growth of the smart card industry.

Although the invention has been described in language specific to structural features and/or methodological steps, it is to be understood that the invention defined in the appended claims is not necessarily limited to the specific features or

steps described. Rather, the specific features and steps are disclosed as preferred forms of implementing the claimed invention.

5

CLAIMS

1. An integrated circuit (IC) card comprising:

an input/output (I/O) interface; and

a smart card development interface, responsive to commands received via the I/O interface, to receive remote procedure calls (RPC) from a computer system executing an application under development within a development application and to invoke an application program interface (API) called in the RPC.

2. An integrated circuit (IC) card according to claim 1, wherein the smart card development interface includes:

a memory having stored therein a plurality of executable instructions; and

control logic, coupled to the memory, to execute the plurality of executable instructions to implement a smart card development interface, wherein the smart card development interface receives and un-marshals a remote procedure call (RPC) from a remote computer system and invokes one of a plurality of application program interface(s) (API) corresponding to the received RPC.

3. An IC card according to claim 2, wherein the control logic establishes a response buffer in an associated memory and marshals a response to the RPC based, at least in part, on the called API.

4. An IC card according to claim 2, wherein the control logic receives the RPC from the remote client computer via an removably coupled IC card interface, responsive to the remote computer

5. An IC card according to claim 2, further comprising a storage medium having stored thereon a plurality of executable instructions which, when executed, implement the smart card development interface.

6. An IC card according to claim 2, wherein the called API's are standard API's of the IC card operating system.

7. An IC card according to claim 2, wherein the called API's are special development API's of the IC card operating system invoked only from a remote computer executing an associated application development program.

8. An IC card according to claim 2, wherein the IC card receives the RPC according to any of a plurality of IC card communication standards.

9. An IC card according to claim 2, wherein the IC card posts a response to the RPC in one of a plurality of IC card communication standards corresponding to the standard in which the RPC was received.

10. An IC card according to claim 2, wherein the IC card receives RPC's from any of a number of remote computers communicatively coupled to the remote computers via a removably coupled IC card interface and a communication network.

11. A computer system comprising:

a memory device having stored therein a plurality of instructions;

a processor, coupled to the memory device, to execute the plurality of instructions and implement a smart card application development tool within which smart card applications are developed with remote procedure call(s) (RPC) to application program interface(s) (API); and

a client development interface, coupled to the memory device and the processor, responsive to the execution of the RPC to marshal and issue a command to a smart card invoking an associated API.

12. A computer system according to claim 11, wherein the client development interface comprises a proxy library containing smart card API information.

13. A computer system according to claim 11, wherein the proxy library includes information regarding the smart card API's including a number and type of parameters associated with each smart card API.

14. A computer system according to claim 11, wherein the smart card application development tool is a common software development application executing on the computer system.

15. A computer system according to claim 11, wherein the client development interface issues an API call to a smart card via a communication channel and a smart card interface.

16. A computer system according to claim 11, wherein the smart card interface is a common card reader.

17. A computer system according to claim 11, wherein the client development interface issues an API call to the smart card according to any of a plurality of standard smart card communication protocols.

18. A computer system according to claim 11, wherein the client development interface issues an API cal to the smart card according to a non-standard smart card development protocol.

19. A computer system according to claim 11, wherein the client development interface receives a response to an API call from the smart card, and posts the response to calling application.

20. A computer system according to claim 19, wherein the calling application is the smart card application development tool.

21. A computer system according to claim 19, wherein the calling application is an application under development within the smart card application development tool.

22. A smart card application development system comprising:

a computer system having a smart card application development tool stored thereon; and

a smart card, responsive to the computer system, including a smart card development interface to receive remote procedure call(s) (RPC) to application program interface(s) (API) associated with the smart card operating system from the smart card application development tool.

23. A smart card application development system according to claim 22, wherein the computer system further comprises a client development interface, to receive and identify the smart card API associated with a particular RPC.

24. A smart card application development system according to claim 23, wherein the client development interface includes a proxy library with information regarding each of a plurality of API calls, including a number and type of parameters required to make the API call.

25. A smart card according to claim 24, wherein the client development interface receives an RPC and marshals the parameters identified in the proxy library before issuing the API call identified within the RPC.

26. A smart card according to claim 24, wherein the client development interface identifies an API associated with an RPC call and issues the API call to the smart card.

27. A smart card development system according to claim 26, wherein the client development interface issues the API call to the smart card via a communication channel and a smart card interface according to any of a plurality of standard smart card communication protocols.

28. A smart card development system according to claim 22, wherein the smart card comprises a smart card development interface to receive API calls from the computer system and to invoke called APIs.

29. A smart card development system according to claim 28, wherein the smart card development interface marshals response parameters to issue to the computer system, depending on a definition of the called API.

30. A smart card development system according to claim 28, wherein the smart card development interface issues a response to the computer system in a select one of a plurality of standard smart card communication protocols corresponding to the smart card communication protocol in which the API call was received.

31. A smart card development system according to claim 22, wherein the smart card application development tool is a common software development application.

32. A method for developing smart card applications comprising:

writing software in any one of a plurality of common software development languages using a computer system; and

embedding within the software remote procedure calls (RPC) to one or more of a plurality of smart card application program interfaces (API) available within a coupled smart card.

33. A method for developing smart card applications comprising:

coupling a computer having stored thereon an application development tool to a card reader with a removably coupled smart card;

executing an application on the computer within the application development tool until a remote procedure call (RPC) to a smart card application program interface (API) is encountered within the application; and

assembling necessary parameters to issue a call to the API identified in the RPC and calling the API at the smart card via the card reader.

34. A method for developing smart card applications according to claim 33, wherein the application development tool is a common software development application.

35. A method for developing smart card applications according to claim 33, further comprising accessing a proxy library within the computer system, the proxy library including information regarding each of a plurality of smart card APIs including a number and type of necessary parameters required to invoke each of the smart card APIs

36. A method for developing smart card applications according to claim 33, wherein the assembling step comprises marshaling necessary parameters for the called API, as defined within a proxy library, in a send buffer.

37. A method for developing smart card applications according to claim 36, further comprising issuing the send buffer to the smart card when all necessary parameters have been marshaled.

38. A method for developing smart card applications according to claim 37, further comprising invoking the called API at the smart card upon receipt of the send buffer.

39. A method for developing smart card applications according to claim 38, further comprising assembling a response buffer at the smart card within which to marshal response parameters for the called API.

40. A method for developing smart card applications according to claim 39, wherein the smart card issues a response to the called API which, when received by the computer system is posted to the executing application.

41. A proxy library comprising:

a plurality of fields containing information regarding a number of smart card application program interfaces (API), wherein the proxy library provides a parameter definition of a smart card application program interface (API) when accessed by control logic upon receiving a remote procedure call (RPC) from an application executing on a computer system, the proxy library enabling the

application executing on the computer system to utilize the resources of the coupled smart card without having to download the application to the smart card.

42. A proxy library according to claim 41, wherein the proxy library is located on a storage medium within the computer system.

43. A proxy library according to claim 41, wherein the proxy library is accessible by a plurality of computers through a communication network.

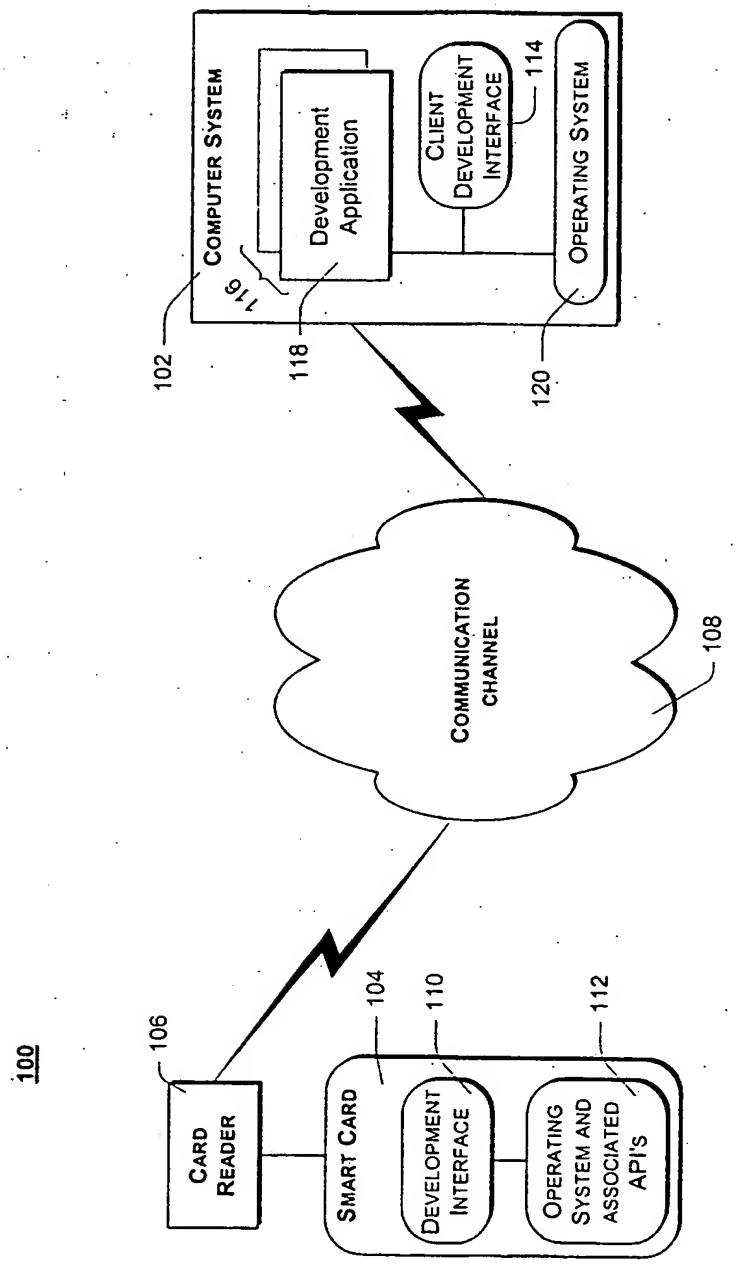

100
COMPUTER SYSTEM
Development Application
CLIENT DEVELOPMENT INTERFACE
OPERATING SYSTEM
102
116
118
114
120
COMMUNICATION CHANNEL
108
106
CARD READER
SMART CARD
DEVELOPMENT INTERFACE
OPERATING SYSTEM AND ASSOCIATED API'S
104
110
112

Fig. 1

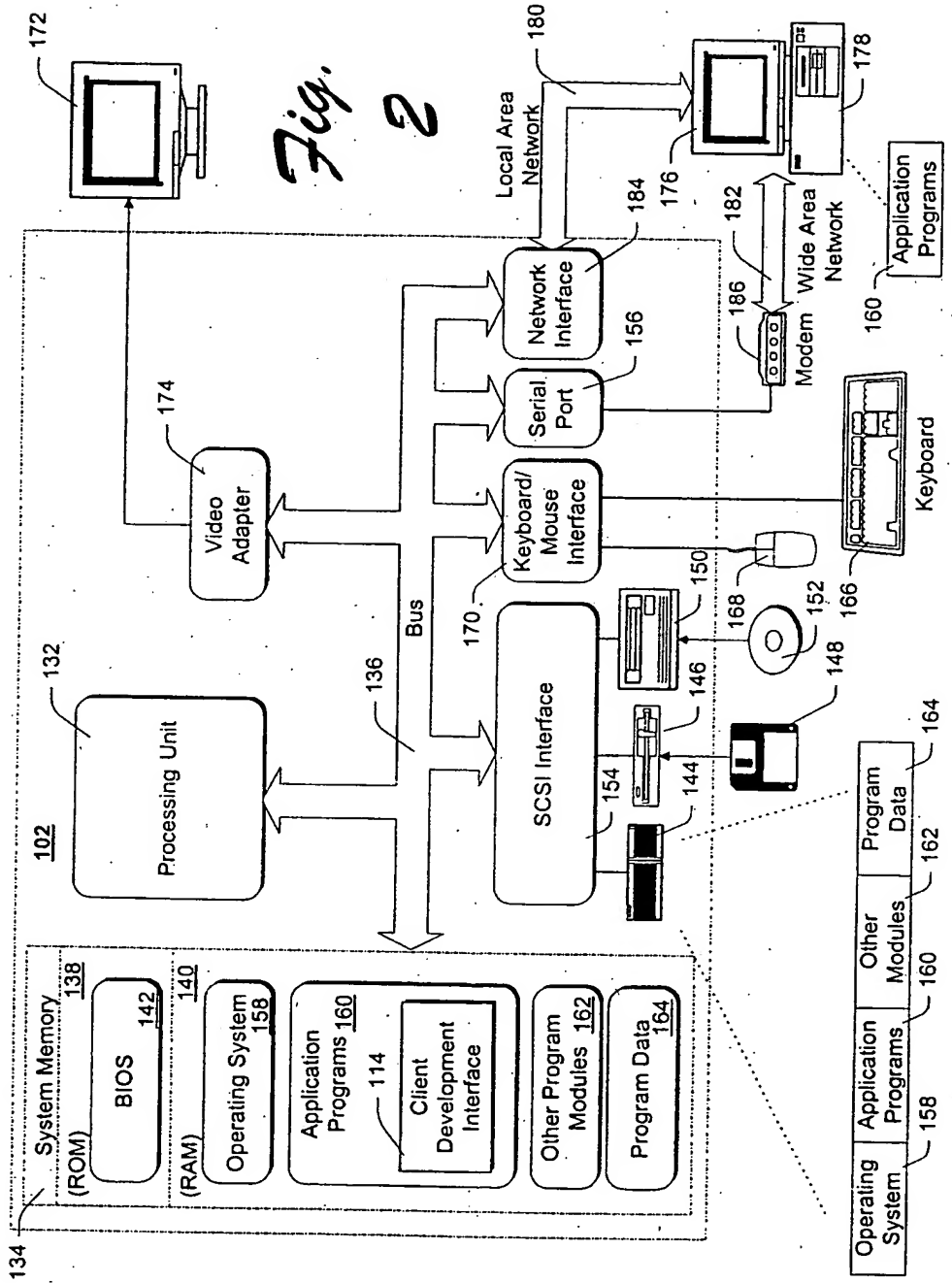


Fig. 2
172
180
Local Area Network
178
176
182
184
Wide Area Network
Application Programs
160
186
Modem
156
Network Interface
Serial Port
174
Video Adapter
Keyboard/ Mouse Interface
Keyboard
150
168
152
166
170
Bus
132
136
Processing Unit
SCSI Interface
148
146
154
144
102
System Memory
138
(ROM)
BIOS
142
140
(RAM)
Operating System
158
Application Programs
160
114
Client Development Interface
Other Program Modules
162
Program Data
164
134
Operating System
158
Application Programs
160
Other Modules
162
Program Data
164

Fig. 4

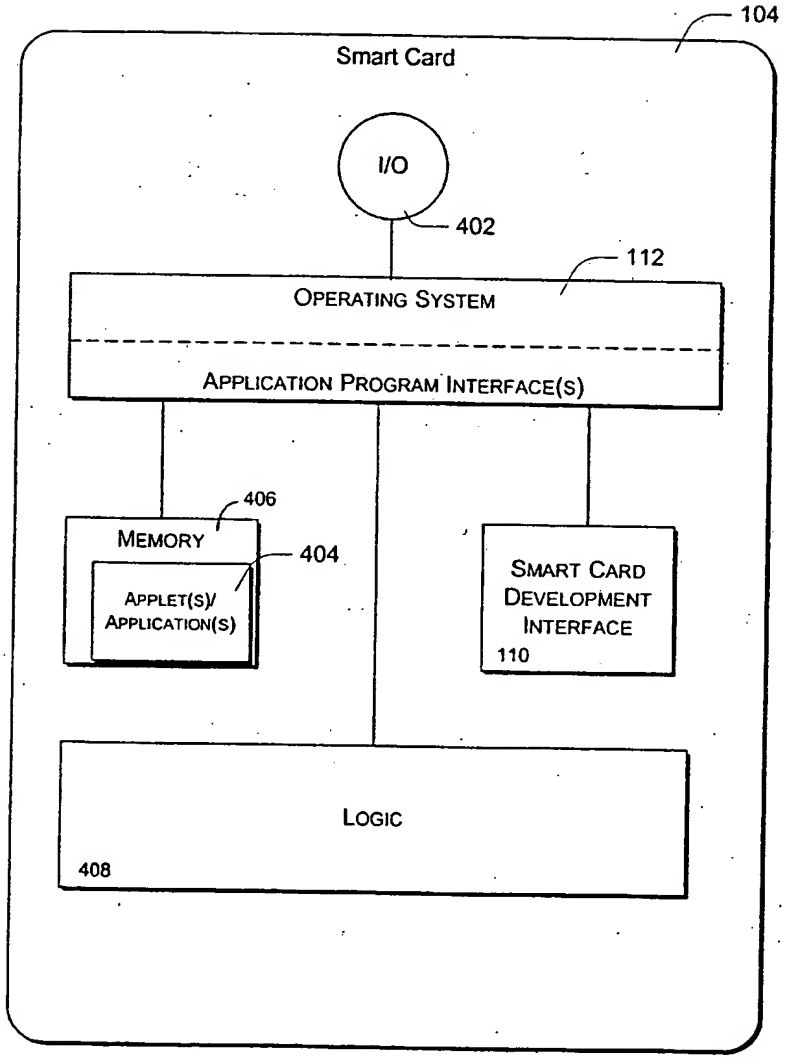

104
Smart Card
I/O
402
112
OPERATING SYSTEM
APPLICATION PROGRAM INTERFACE(S)
406
MEMORY
404
APPLET(S)/
APPLICATION(S)
SMART CARD
DEVELOPMENT
INTERFACE
110
LOGIC
408

Fig. 3

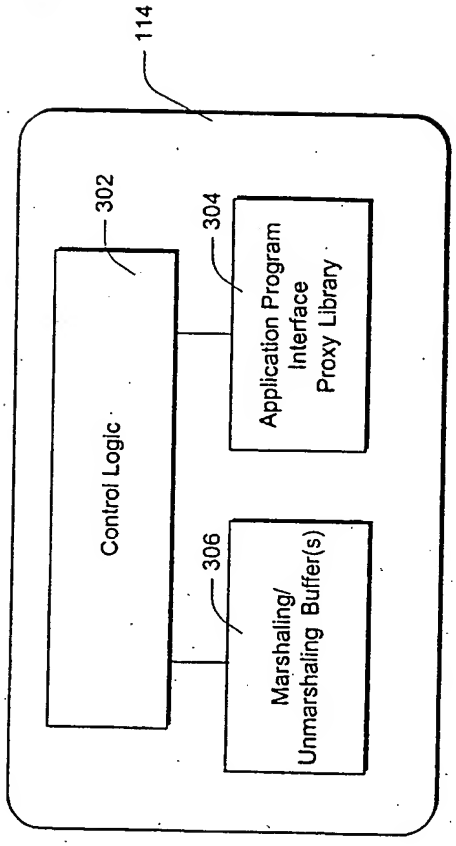

114
302
304
306
Control Logic
Application Program Interface Proxy Library
Marshaling/ Unmarshaling Buffer(s)

Fig. 5

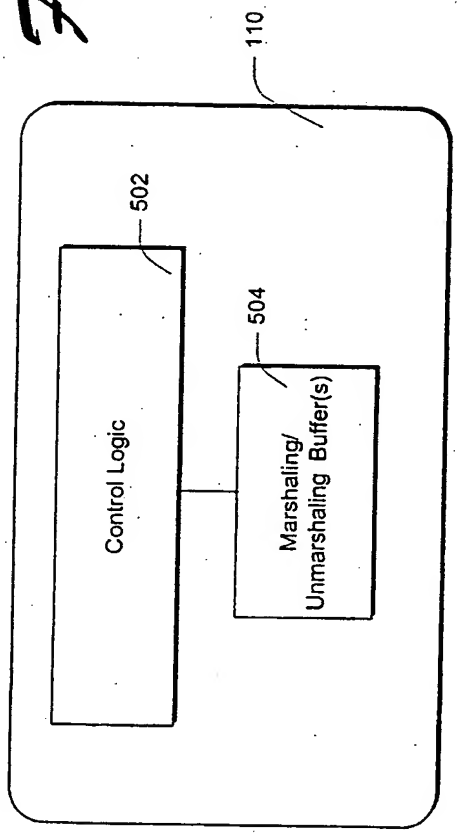

110
502
504
Control Logic
Marshaling/ Unmarshaling Buffer(s)

Fig. 6

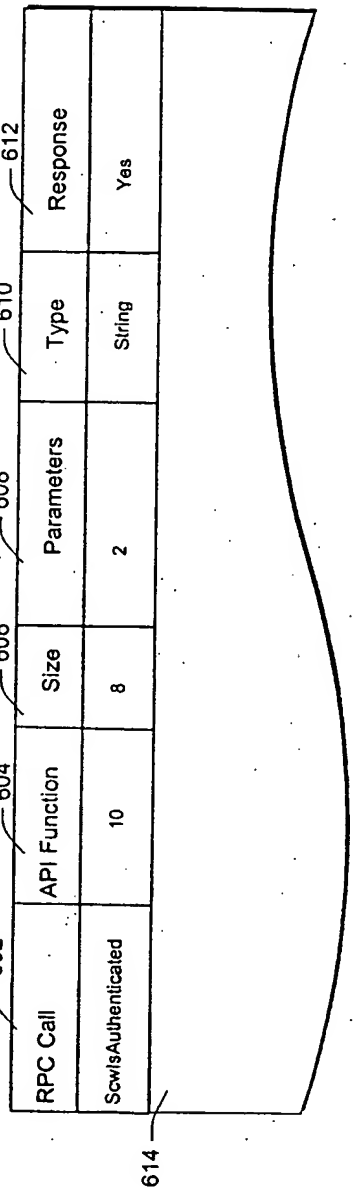

600
602
604
606
608
610
612
614
RPC Call
API Function
Size
Parameters
Type
Response
ScwIsAuthenticated
10
8
2
String
Yes

Fig. 7

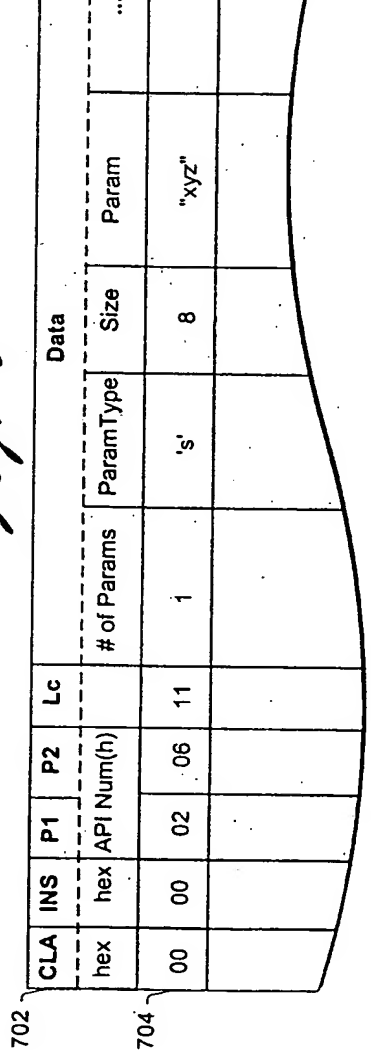

306
702
704
CLA
INS
P1
P2
Lc
Data
hex
hex
API Num(h)
# of Params
ParamType
Size
Param
...
00
00
02
06
11
1
's'
8
"xyz"

Fig. 8

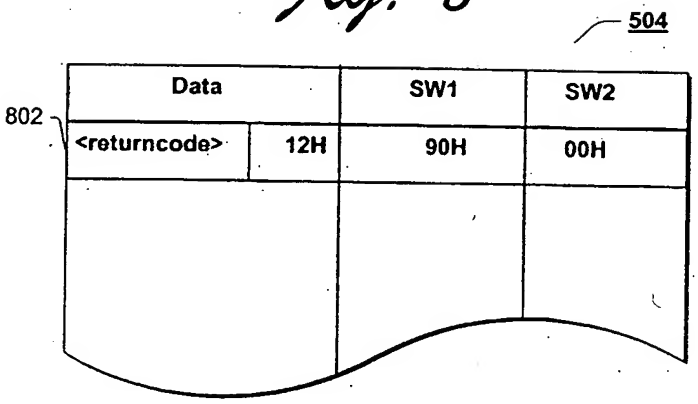
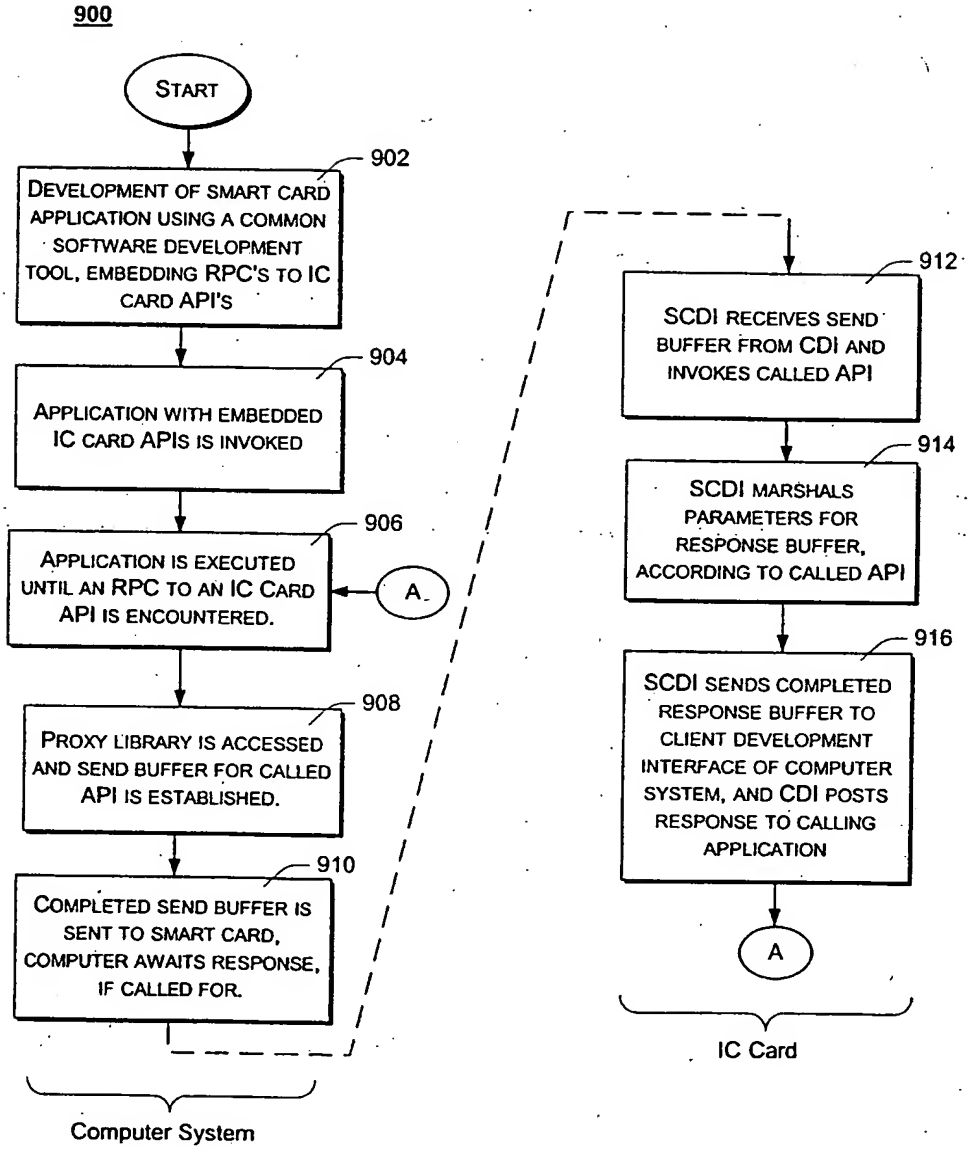


Fig. 9


900
START
902
DEVELOPMENT OF SMART CARD APPLICATION USING A COMMON SOFTWARE DEVELOPMENT TOOL, EMBEDDING RPC'S TO IC CARD API'S
904
APPLICATION WITH EMBEDDED IC CARD APIS IS INVOKED
906
APPLICATION IS EXECUTED UNTIL AN RPC TO AN IC CARD API IS ENCOUNTERED.
A
908
PROXY LIBRARY IS ACCESSED AND SEND BUFFER FOR CALLED API IS ESTABLISHED.
910
COMPLETED SEND BUFFER IS SENT TO SMART CARD, COMPUTER AWAITS RESPONSE, IF CALLED FOR.
Computer System
912
SCDI RECEIVES SEND BUFFER FROM CDI AND INVOKES CALLED API
914
SCDI MARSHALS PARAMETERS FOR RESPONSE BUFFER, ACCORDING TO CALLED API
916
SCDI SENDS COMPLETED RESPONSE BUFFER TO CLIENT DEVELOPMENT INTERFACE OF COMPUTER SYSTEM, AND CDI POSTS RESPONSE TO CALLING APPLICATION
A
IC Card

INTERNATIONAL SEARCH REPORT

In rtional Application No
PCT/US 00/12933

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G07F7/10

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G07F G06F H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
WPI Data, EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 98 09257 A (GEMPLUS) 5 March 1998 (1998-03-05) abstract; claims; figures 1-4 page 12, line 1 -page 18, line 2	1,2,4,6, 11, 15-17, 22,32,33
A	WO 98 19237 A (SCHLUMBERGER TECHNOLOGIES) 7 May 1998 (1998-05-07) abstract; claims; figures page 16, line 5 - line 22	1,11,22, 32,33,41
A	WO 98 25239 A (STRATEGIC ANALYSIS) 11 June 1998 (1998-06-11) abstract; claims; figures	1,11,22, 32,33,41
A	WO 98 43212 A (VISA INTERNATIONAL SERVICE ASSOCIATION) 1 October 1998 (1998-10-01)	

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
29 September 2000	10/10/2000

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	David, J

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Int tional Application No
PCT/US 00/12933

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	WO 98 40818 A (MACRONIX INTERNATIONAL) 17 September 1998 (1998-09-17)	

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

Int tional Application No
PCT/US 00/12933

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9809257 A	05-03-1998	US 5923884 A	13-07-1999
		AU 4842897 A	19-03-1998
		CA 2233217 A	05-03-1998
		EP 0858644 A	19-08-1998
WO 9819237 A	07-05-1998	AU 722463 B	03-08-2000
		AU 4911897 A	22-05-1998
		EP 0932865 A	04-08-1999
WO 9825239 A	11-06-1998	AU 5595398 A	29-06-1998
		EP 0943136 A	22-09-1999
WO 9843212 A	01-10-1998	AU 6578698 A	20-10-1998
		EP 1004992 A	31-05-2000
		EP 1021801 A	26-07-2000
		US 6005942 A	21-12-1999
WO 9840818 A	17-09-1998	US 5901330 A	04-05-1999
		EP 0974089 A	26-01-2000